IT-Security (ITS) B1

DIKU, E2022

Today’s agenda

Forensics defined

Disk forensics

Memory forensics

Forensics defined

Digital forensics is a branch of forensic science encompassing the recovery and investigation of material found on digital devices

Applied in a **corporate**, **civil**, or **criminal** setting (originated in law enforcement)

Applied to a **security** investigation (of an intrusion) or a **personnel** investigation

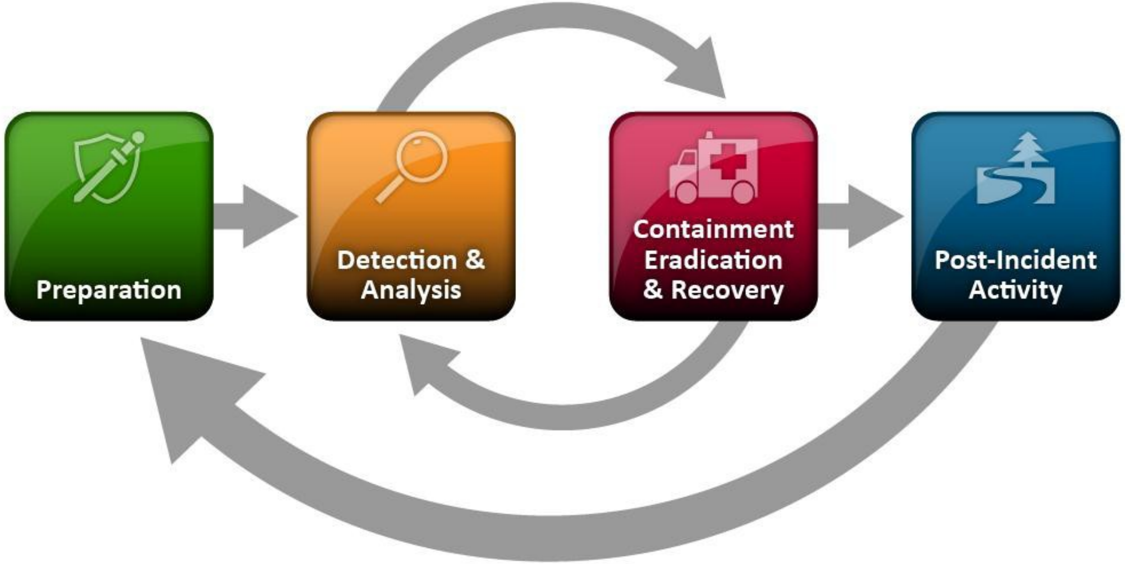
In security investigations, forensics either means a **root cause** or **impact analysis** of a cyber-attack, often post-mortem, **or simply techniques** used in the process of uncovering, understanding, and responding to a security incident

In security, **DFIRMA** = digital forensics + incident response + malware analysis

DFIRMA in practice

```
while true:
    intrusion analysis

    if intrusion suspected:
        preliminary analysis

        if intrusion verified:
            repeat until incident fully grasped:
                incident analysis
                forensic analysis
                malware anaysis

            incident response

    update plans
```

Recap: Intrusion detection



Many forms of forensics

Digital forensics =

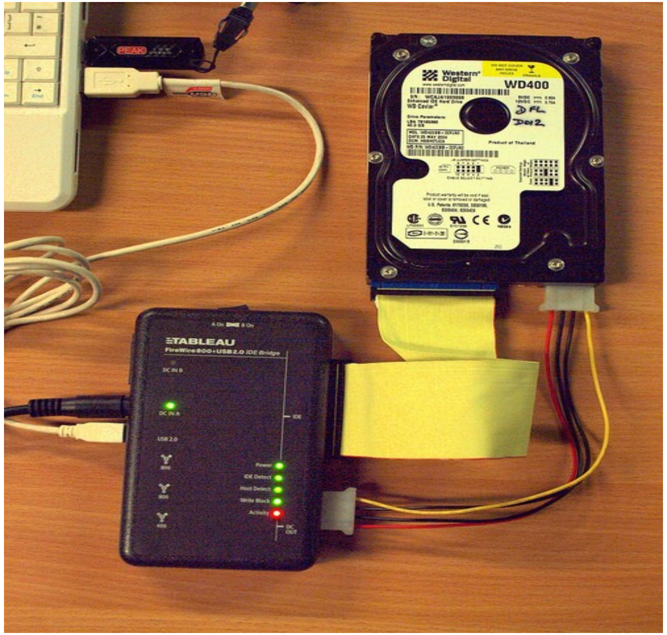
- Computer forensics
- Memory forensics

Our focus today

- Network forensics
- Mobile forensics
- Etc. forensics

Disk (or, file system) forensics

Classic disk forensic approach



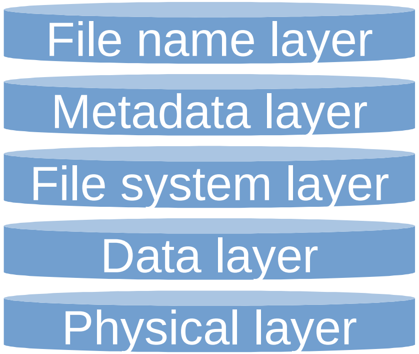
Forensic workstation

Seized harddrive

Write blocker

Forensics in a nutshell

Understanding the low-level details



File name layer

Metadata layer

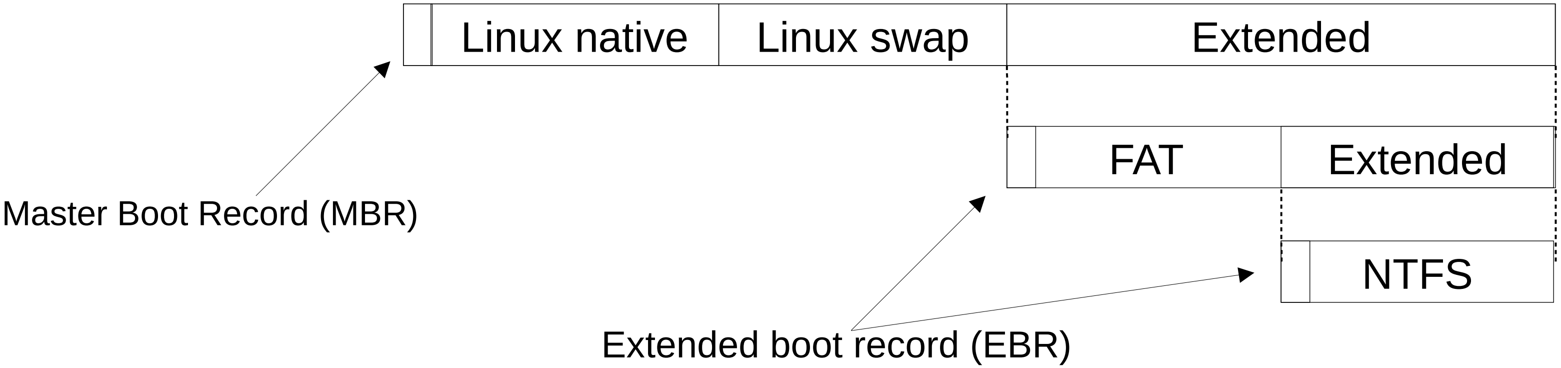
File system layer

Data layer

Physical layer

- File names, directories
- Structure information about files/directories
- Partition information
- Sectors, blocks, clusters
- The drive itself, and partitions

Disk forensic example: DOS partitions



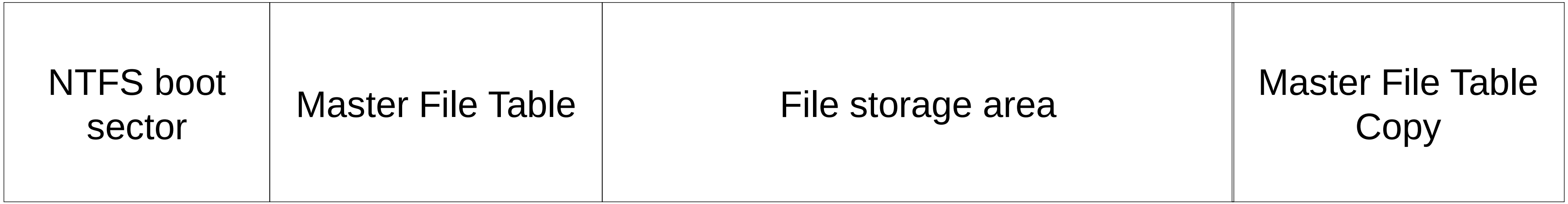
MBR/EBR same layout

Bytes	Content
0-445	Upstart code, disk signature
446-461	Partition entry 1
462-477	Partition entry 2
478-493	Partition entry 3
494-509	Partition entry 4
510-511	MBR/EBR signature (0xAA55)

Bytes	Content
0	0x00 not boot, 0x80 boot
1-3	Cylinder-head-sector (CHS) of start sector
4	Partition type
5-7	Cylinder-head-sector (CHS) of end sector
8-11	Logical block addressing (LBA) of start sector
12-15	Number of sectors in partition

Type	FAT12	FAT16	FAT32	Linux native	Linux swap	Extended	NFTS
Hex value	0x01	0x0E	0x0C	0x83	0x82	0x05	0x07

File system example: NTFS



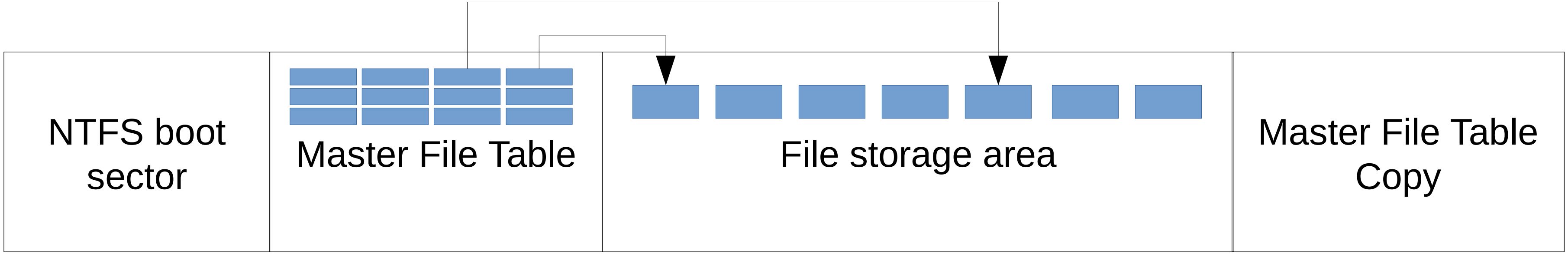
NTFS boot sector

Master File Table

File storage area

Master File Table Copy

File system example: NTFS



Master File Table (MFT)

An entry in the MFT describes a file

- Filename and metadata like permissions, timestamps
- Entries are 1024 bytes
- For larger files (non-resident files), the MFT entry contains links to areas of the disk where the file data resides

Data / File storage area

Clusters (Windows) or **blocks** (Unix) = 1 or more 512-byte **sectors**

Custers/blocks either **allocated**

- Actively being used by a file

Or **unallocated**

- Not being used by a file
- May contain deleted or unused data

Deleted != destroyed

When a file is deleted, **data still exists** on disk until overwritten

If overwritten, **remnants may still exist** in

- extra copies of the file
- page/swap/hibernation file, or
- elsewhere on the disk due to (de)fragmentation

However, if disk wiped, only just once, recovery infeasible

Think libraries

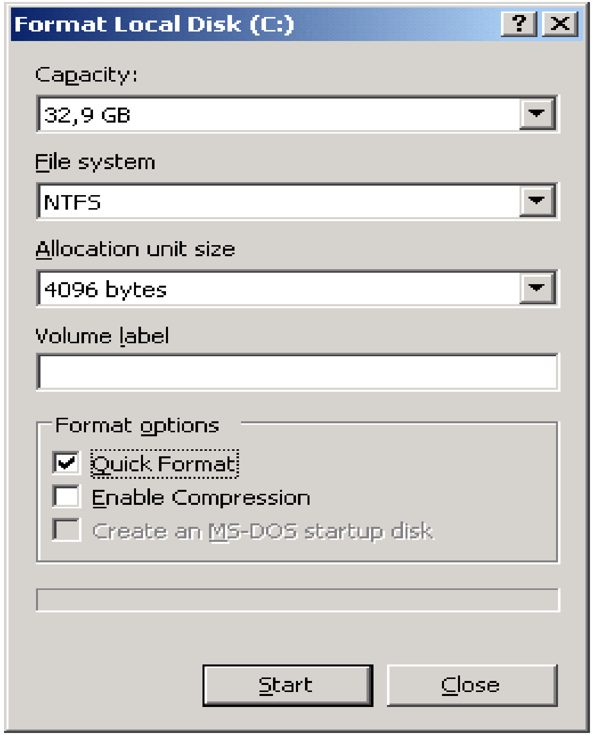
Format is not wiping

Formats create and replace file system structures

Files are not overwritten

Regular formats take longer as the disk is scanned for bad sectors

Use wiping software for wiping



Capacity: 32,9 GB

File system: NTFS

Allocation unit size: 4096 bytes

Volume label:

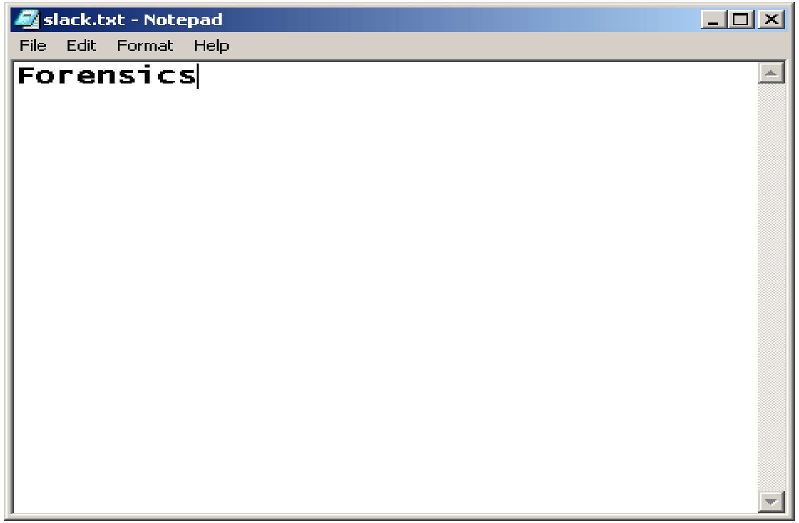
Format options

☑ Quick Format

□ Enable Compression

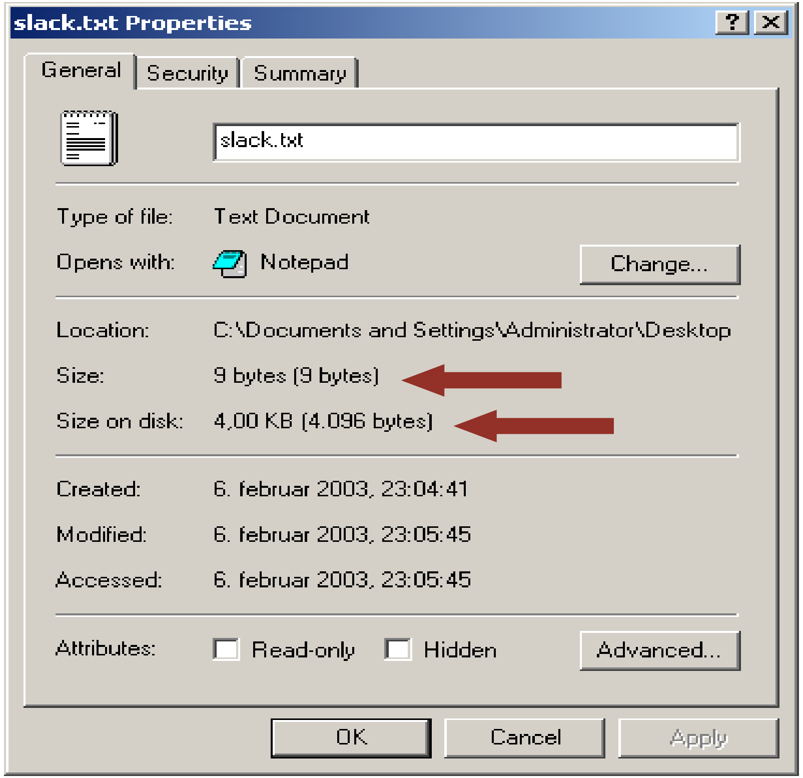
□ Create an MS-DOS startup disk

Slack space



slack.txt - Notepad

Forensics



slack.txt Properties

slack.txt

Type of file: Text Document

Opens with: Notepad

Location: C:\Documents and Settings\Administrator\Desktop

Size: 9 bytes (9 bytes)

Size on disk: 4,00 KB (4.096 bytes)

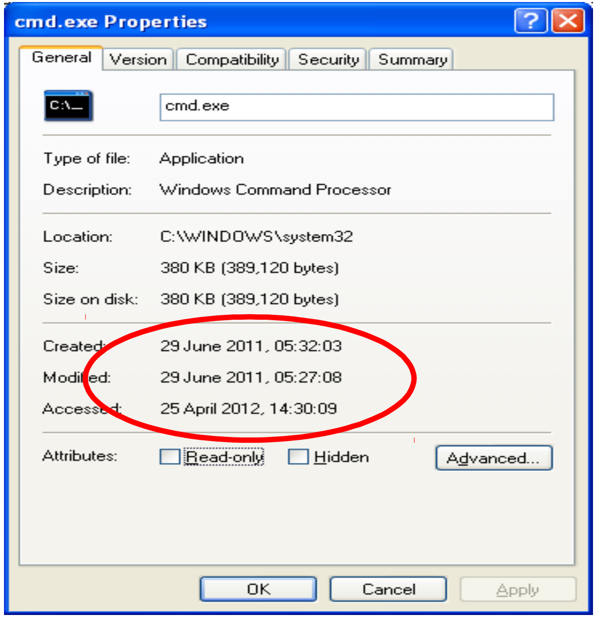
Created: 6. februar 2003, 23:04:41

Modified: 6. februar 2003, 23:05:45

Accessed: 6. februar 2003, 23:05:45

Attributes: □ Read-only □ Hidden

Timeline (Modified, Accessed, Changed)



cmd.exe

Type of file: Application

Description: Windows Command Processor

Location: C:\WINDOWS\system32

Size: 380 KB (389,120 bytes)

Size on disk: 380 KB (389,120 bytes)

Created: 29 June 2011, 05:32:03

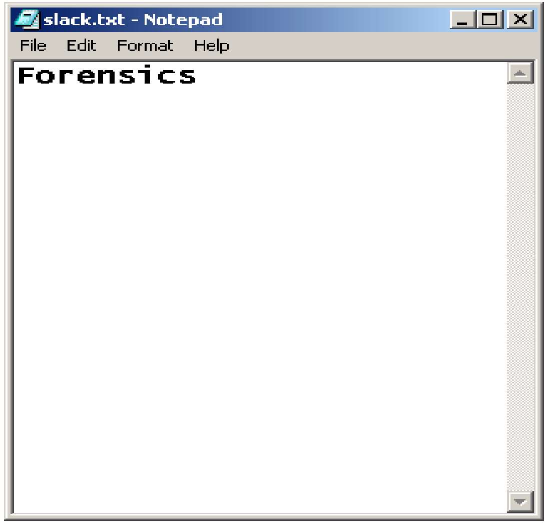
Modified: 29 June 2011, 05:27:08

Accessed: 25 April 2012, 14:30:09

Attributes: □ Read-only □ Hidden

Searching for file types

Slack.txt



Forensics

Slack.exe

Slack.pdf

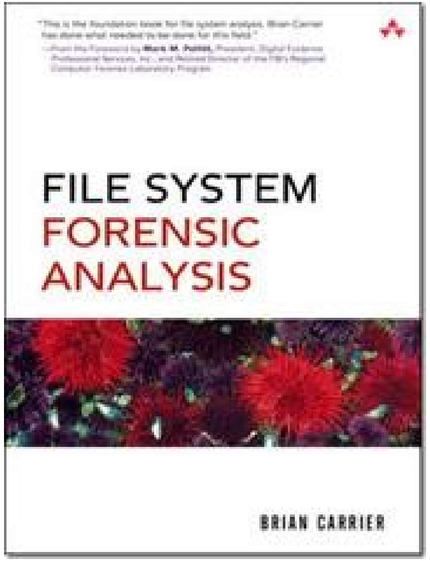
Slack.zip

Slack.dat

Slack.mp3

Slack.dll

Further reading


"This is the foundation book for file system analysis. Brian Carrier has done what needed to be done for this field."
—From the Foreword by Mark M. Pollitt, President, Digital Evidence Professional Services, Inc., and Retired Director of the FBI's Regional Computer Forensic Laboratory Program
FILE SYSTEM FORENSIC ANALYSIS
BRIAN CARRIER

Memory forensics

Memory forensics

From Wikipedia:

“Memory forensics is forensic analysis of a computer's **memory dump**.

Its primary application is investigation of advanced computer attacks which are stealthy enough to avoid leaving data on the computer's hard drive.”

First, get a copy

- Live acquisition
    - Different techiniques
- Live analysis
    - Direct analysis of the running kernel
- Dead acquisition
    - Hibernation files, page files
- Virtualization - thank you

What to find in memory?

Running processes

Listening sockets

Open connections

Encryption keys

Credentials

Memory only malware

Closed connections

Terminated processes

Open file handles

Deobfuscated code

Memory forensic analysis process

1: Find rogue processes

2: Analyse DLLs

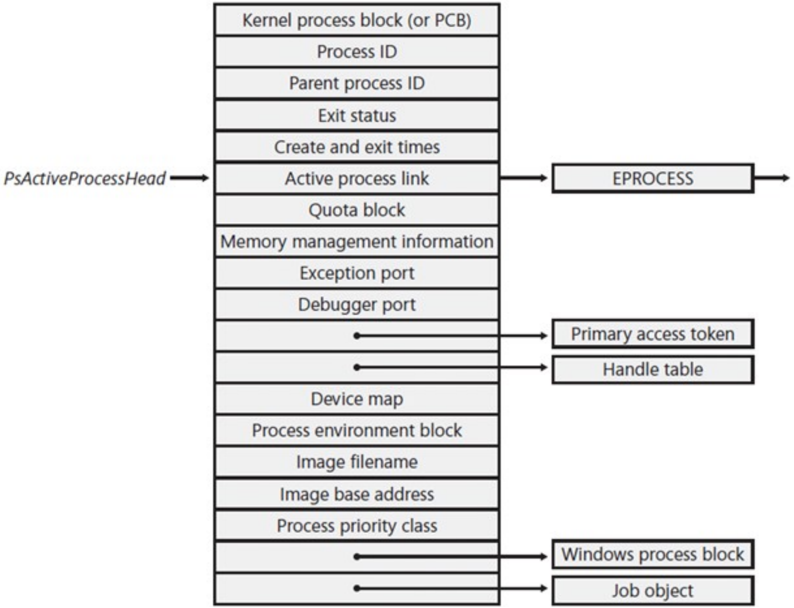
3: Review network artefacts

4: Look for evidence of code injections

5: Dump suspicious processes → further analysis

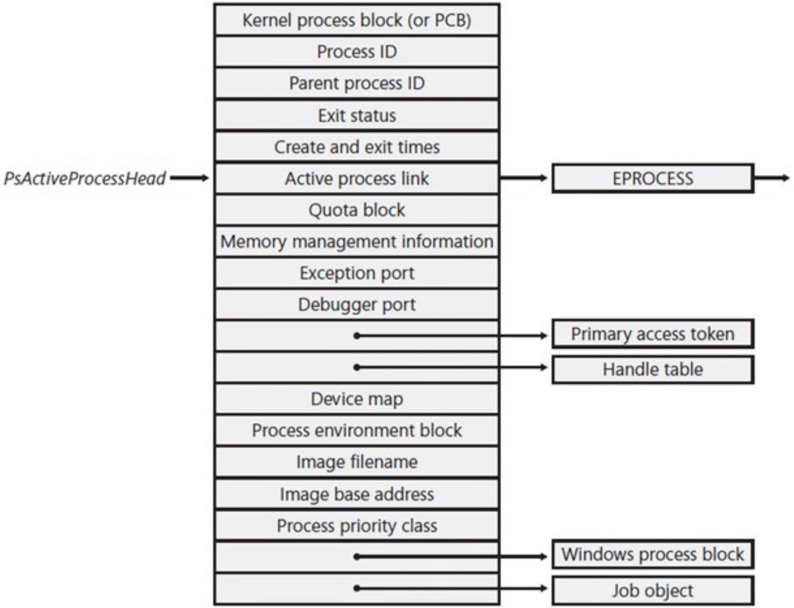
How to find processes (on Windows)

EPROCESS objects in memory:

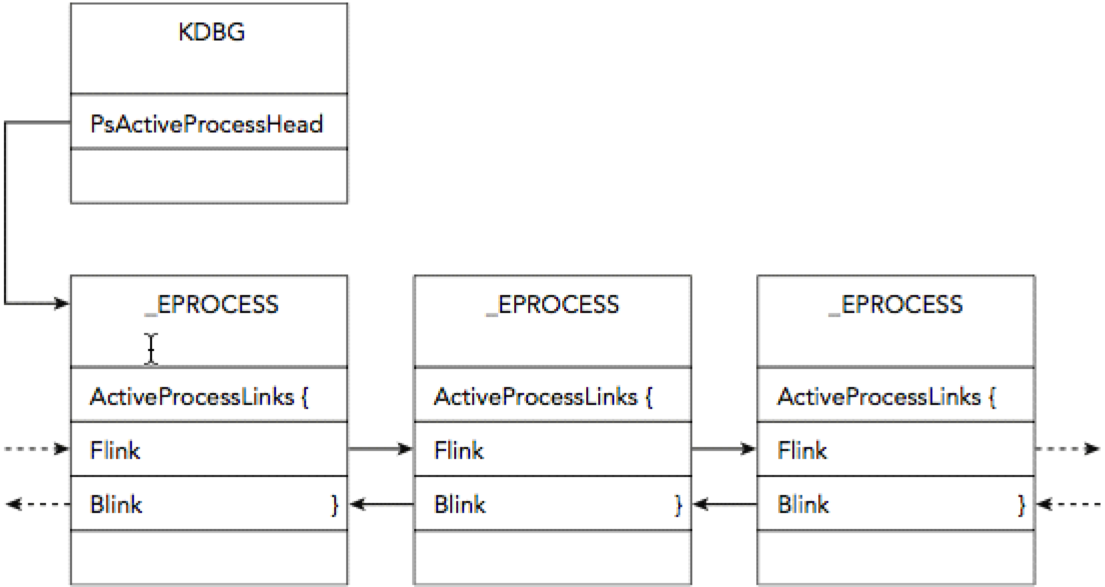


How to find processes (on Windows)

Scan for EPROCESS objects:



Process enumeration (on Windows)



Key concept in memory forensics:

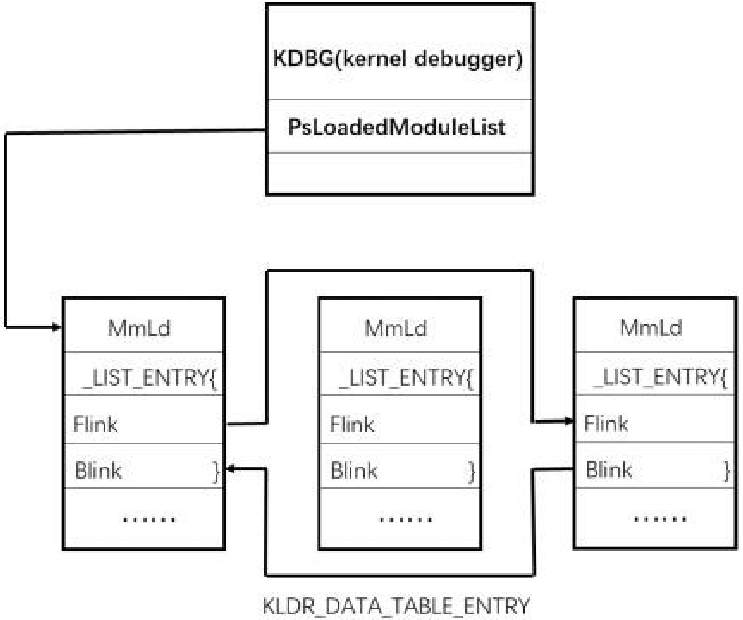
Walking a list, or **scanning for objects**

Step 1 revisited: Find rogue processes

Those that:

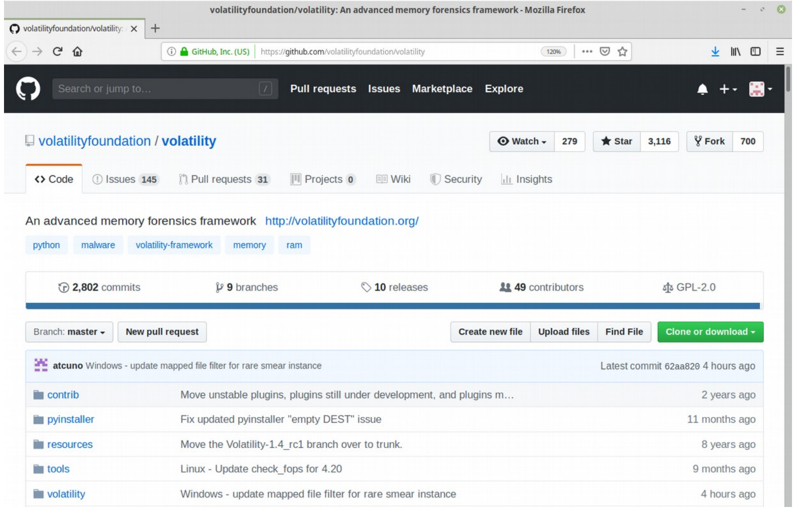
- Hide
- Have odd parents
- Do network comm but shouldn’t
- Have unusually many handles open
- Contain maliciously injected code
- ...

Direct kernel objection manipulation (DKOM)



Volatility

Volatility is an open source memory analysis framework writtin in Python



volatilityfoundation/volatility: An advanced memory forensics framework - Mozilla Firefox

https://github.com/volatilityfoundation/volatility

volatilityfoundation / volatility

An advanced memory forensics framework http://volatilityfoundation.org/

2,802 commits | **9** branches | **10** releases | **49** contributors | GPL-2.0

atcuno Windows - update mapped file filter for rare smear instance — Latest commit 62aa820 4 hours ago

contrib	Move unstable plugins, plugins still under development, and plugins m…	2 years ago
pyinstaller	Fix updated pyinstaller "empty DEST" issue	11 months ago
resources	Move the Volatility-1.4_rc1 branch over to trunk.	8 years ago
tools	Linux - Update check_fops for 4.20	9 months ago
volatility	Windows - update mapped file filter for rare smear instance	4 hours ago

Example:

Stuxnet

Stuxnet

DO WE HAVE A BACKUP?
Stuxnet VIRUS
CHAPPATTE
Int'l Herald Tribune

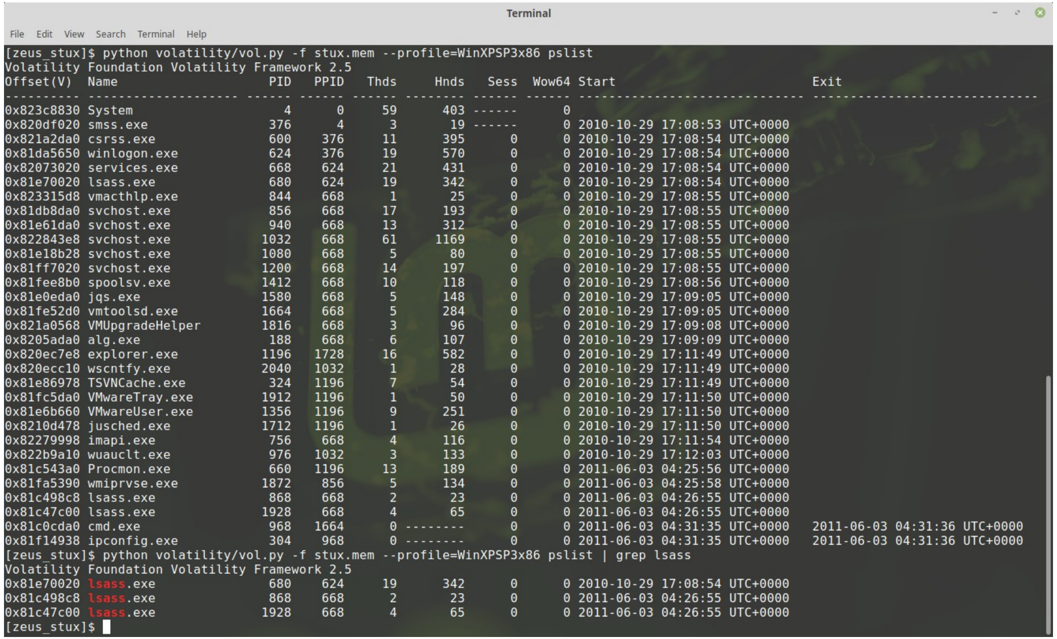
Stuxnet

Natanz Nuclear Facility in Iran

Stuxnet

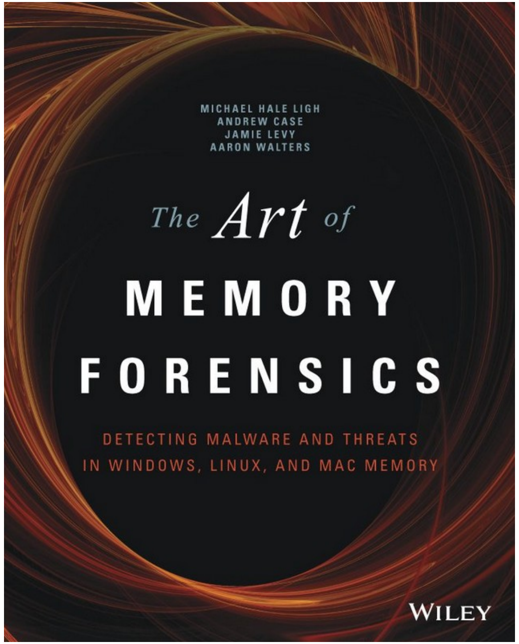
STUXNET: The Virus that Almost Started WW3

Volatility and Stuxnet



```
[zeus_stux]$ python volatility/vol.py -f stux.mem --profile=WinXPSP3x86 pslist
Volatility Foundation Volatility Framework 2.5
Offset(V)  Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start                          Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ ------------------------------ ------------------------------
0x823c8830 System                    4      0     59      403 ------      0
0x820df020 smss.exe                376      4      3       19 ------      0 2010-10-29 17:08:53 UTC+0000
0x821a2da0 csrss.exe               600    376     11      395      0      0 2010-10-29 17:08:54 UTC+0000
0x81da5650 winlogon.exe            624    376     19      570      0      0 2010-10-29 17:08:54 UTC+0000
0x82073020 services.exe            668    624     21      431      0      0 2010-10-29 17:08:54 UTC+0000
0x81e70020 lsass.exe               680    624     19      342      0      0 2010-10-29 17:08:54 UTC+0000
0x823315d8 vmacthlp.exe            844    668      1       25      0      0 2010-10-29 17:08:55 UTC+0000
0x81db8da0 svchost.exe             856    668     17      193      0      0 2010-10-29 17:08:55 UTC+0000
0x81e61da0 svchost.exe             940    668     13      312      0      0 2010-10-29 17:08:55 UTC+0000
0x822843e8 svchost.exe            1032    668     61     1169      0      0 2010-10-29 17:08:55 UTC+0000
0x81e18b28 svchost.exe            1080    668      5       80      0      0 2010-10-29 17:08:55 UTC+0000
0x81ff7020 svchost.exe            1200    668     14      197      0      0 2010-10-29 17:08:55 UTC+0000
0x81fee8b0 spoolsv.exe            1412    668     10      118      0      0 2010-10-29 17:08:56 UTC+0000
0x81e0eda0 jqs.exe                1580    668      5      148      0      0 2010-10-29 17:09:05 UTC+0000
0x81fe52d0 vmtoolsd.exe           1664    668      5      284      0      0 2010-10-29 17:09:05 UTC+0000
0x821a0568 VMUpgradeHelper        1816    668      3       96      0      0 2010-10-29 17:09:08 UTC+0000
0x8205ada0 alg.exe                 188    668      6      107      0      0 2010-10-29 17:09:09 UTC+0000
0x820ec7e8 explorer.exe           1196   1728     16      582      0      0 2010-10-29 17:11:49 UTC+0000
0x820ecc10 wscntfy.exe            2040   1032      1       28      0      0 2010-10-29 17:11:49 UTC+0000
0x81e86978 TSVNCache.exe           324   1196      7       54      0      0 2010-10-29 17:11:49 UTC+0000
0x81fc5da0 VMwareTray.exe         1912   1196      1       50      0      0 2010-10-29 17:11:50 UTC+0000
0x81e6b660 VMwareUser.exe         1356   1196      9      251      0      0 2010-10-29 17:11:50 UTC+0000
0x8210d478 jusched.exe            1712   1196      1       26      0      0 2010-10-29 17:11:50 UTC+0000
0x82279998 imapi.exe               756    668      4      116      0      0 2010-10-29 17:11:54 UTC+0000
0x822b9a10 wuauclt.exe             976   1032      3      133      0      0 2010-10-29 17:12:03 UTC+0000
0x81c543a0 Procmon.exe             660   1196     13      189      0      0 2011-06-03 04:25:56 UTC+0000
0x81fa5390 wmiprvse.exe           1872    856      5      134      0      0 2011-06-03 04:25:58 UTC+0000
0x81c498c8 lsass.exe               868    668      2       23      0      0 2011-06-03 04:26:55 UTC+0000
0x81c47c00 lsass.exe              1928    668      4       65      0      0 2011-06-03 04:26:55 UTC+0000
0x81c0cda0 cmd.exe                 968   1664      0 --------      0      0 2011-06-03 04:31:35 UTC+0000   2011-06-03 04:31:36 UTC+0000
0x81f14938 ipconfig.exe            304    968      0 --------      0      0 2011-06-03 04:31:35 UTC+0000   2011-06-03 04:31:36 UTC+0000
[zeus_stux]$ python volatility/vol.py -f stux.mem --profile=WinXPSP3x86 pslist | grep lsass
Volatility Foundation Volatility Framework 2.5
0x81e70020 lsass.exe               680    624     19      342      0      0 2010-10-29 17:08:54 UTC+0000
0x81c498c8 lsass.exe               868    668      2       23      0      0 2011-06-03 04:26:55 UTC+0000
0x81c47c00 lsass.exe              1928    668      4       65      0      0 2011-06-03 04:26:55 UTC+0000
[zeus_stux]$
```

Further reading



MICHAEL HALE LIGH
ANDREW CASE
JAMIE LEVY
AARON WALTERS

The Art of

MEMORY FORENSICS

DETECTING MALWARE AND THREATS
IN WINDOWS, LINUX, AND MAC MEMORY

WILEY